Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 60 (2017) 128 – 132

www.elsevier.com/locate/procedia

27th CIRP Design 2017

Framework for Engineering Design Systems Architectures Evaluation and Selection: Case Study

Mohamed Darwish[a*], Essam Shehab[a]

[a]School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield, Bedford MK43 0AL, UK

* Corresponding author. Tel.: +44 (0)1234750111. *E-mail address:* m.a.darwish@cranfield.ac.uk

Abstract

Engineering companies face the challenge of developing complex Engineering Design Systems. These systems involve huge financial, people, and time investments within an environment that is characterised by continuously changing technologies and processes. Systems architecture provides the strategies and modelling approaches to ensure that adequate resources are spent in developing the possible To Be states for a target system. Architecture selection and evaluation involves evaluating different architectural alternatives with respect to multiple criteria, hence an Architecture Evaluation Framework which evaluates and down selects the appropriate architectures solutions is crucial to assess how these systems will deliver value over their lifetime, and where to channel the financial and human investments to maximize benefit delivered to the business' bottom line.

In this paper, an evaluation and selection architecture framework is proposed, which targets to maximise the alignment of Engineering Design Systems with business goals based on a quality centric architecture evaluation approach. The framework utilised software Quality Attributes as well as SWOT (Strength, Weakness, Opportunity, Threat) and PEST (Political, Economic, Social, Technological) analyses to capture different viewpoints related to technical, political and business context. The framework proposed employing AHP (Analytical Hierarchy Process) to quantitatively elicit relationships between Quality Attributes trade-offs and architectural characteristics. The framework was applied to a real case study considering five Engineering Design Systems alternative architectures, where workshops with subject matter experts and stakeholders were held to reach an informative decision, that maximise architectural quality, whilst maintaining business alignment.

Peer-review under responsibility of the scientific committee of the 27th CIRP Design Conference

Keywords: Engineering design system architecutre, architecture evaluation, quality attributes, analytic hierarchy process (AHP), SWOT and PEST

1. Introduction

Engineering companies face the challenge of developing complex engineering design systems. These systems involve huge financial, people, and time investments within an environment that is characterised by continuously changing technologies and processes. Systems architecting provides the strategies and modelling approaches to ensure that adequate resources are spent in developing the possible To BE states for a target system. Architecture evaluation involves evaluating different architecture alternatives with respect to multiple criteria, hence a rigorous Architecture Evaluation Framework to evaluate architectural alternatives is crucial to assess how these systems will deliver value over their lifetime, and where to channel the financial and human investments to maximize the benefit to the businesses bottom line.

This paper gives an overview of the theoretical background of evaluation processes and Quality Attributes trade-offs and highlights the importance of appreciating business context of engineering systems when evaluating alternative solutions.

An evaluation and selection architecture framework is proposed, based on a quality centric architecture evaluation approach. Analytical Hierarchy Process (AHP) is utilised to quantitatively elicit relationship between Quality Attributes trade-offs and architecture characteristics. The Quality Attributes utilised are adopted from ISO/IEC 25010:2011 standard. The framework also employs SWOT and PEST

Peer-review under responsibility of the scientific committee of the 27th CIRP Design Conference
doi:10.1016/j.procir.2017.01.058

analyses to capture different viewpoints related to political, societal and business contexts.

The framework was applied to a real case study considering five alternative architectures. Data collected has been analysed by a commercial AHP tool. The results, together with workshops discussion, have assisted stakeholders to reach an informative decision.

Nomenclature

AHP	Analytical Hierarchy Process
API	Application Programming Interface
DSL	Domain Specific Language
MCDM	Multi-Criteria Decision Making
MDE	Model Driven Engineering
PEST	Political, Economic, Social, Technological
QA	Quality Attributes
SQuaRE	Software Quality Requirements and Evaluation
SWOT	Strength, Weakness, Opportunity, Threat

2. Literature review and theoretical background

Engineering products and systems are becoming increasingly complex, not only driven by global competition and price pressure, but also with fast moving customers' requirements [1]. High level of complexity and customers' changes cause systems to grow over time in order to increase capabilities, hence leading to having evolved Engineering Design Systems and Sub-Systems that are not designed to support scalability. Instead, they were designed to meet specific and timely needs [2].

Systems architecting provides the strategies and modelling approaches to ensure that adequate resources is spent in developing the possible 'could be' states, and evaluating and selecting the best alternative given a set of desired properties and criteria for the future system [3]. As Design Systems become larger and more complex, their architectures assume ever greater importance in managing their growing integrity and coherence. Thus, when architectural integrity is compromised, the probability for serious operational problems increases dramatically. Interactions among layers and subsystems become increasingly more difficult to understand. The ability to assess unwanted side effects before implementing changes becomes more laborious. Modifications will be more intricate and tedious. Consequently, the verification of functional and structural quality becomes less thorough when speed delivery is the priority. Thus, architectural integrity enables safe rapid development cycles whilst maintain quality and safety [4].

2.1. .System Architecture Quality Attributes

Functional requirements show the ability of the system to deliver the services which it was designed for. However, how well the system caters for modifications like scalability, maintainability or portability is best assessed through capturing Quality Attributes (non-functional requirements), which are properties of a system that are used to indicate how well the system satisfies the needs of its stakeholders for future change [5].

Several Quality Models that provide hierarchical order of Quality Attributes have been published in the last decades [6]. One of the earliest models was established by Boehm et al. to define software quality through a given set of attributes and metrics [7]. Later models were defined through international standards such as ISO/IEC 9126-1:2001 [Software engineering Product quality], which was later revised by ISO/IEC 25010:2011 [Systems and software engineering, Systems and software Quality Requirements and Evaluation (SQuaRE)] [8].

ISO/IEC 25010:2011 standard classifies software quality within taxonomy of characteristics and sub-characteristics. The characteristics considered are; functionality, reliability, usability, efficiency, maintainability, and portability. Each of these characteristics is subdivided into Quality Attributes (Fig. 1) that can be measured and verified [4].

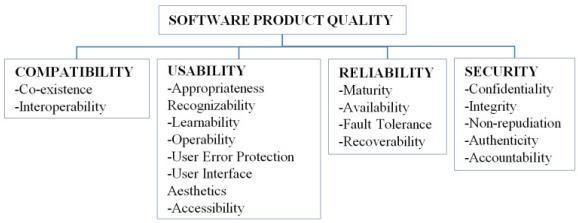

Fig. 1. Subset of ISO/IEC 25010:2011 Quality Model [8]

2.2. Systems Architecture Quality Attributes Trade-offs

A quality-based architecture is one designed to satisfy a single or multiple Quality Attributes. In most cases, it is impossible to maximize all of them, hence the architect must consider a trade-off to ensure high priority functions are not being compromised [9].

Systematic research suggests that there is an immaturity in the field of software quality trade-off, hence no approach or set of approaches have emerged as candidates to dominate the research space, however empirical evidences suggest that Analytical Hierarchy Process (AHP) is one of the most widely applied approach as Multi-Criteria Decision Making (MCDM) tool [10].

AHP is comprised of four main steps [11]:

1) Define the problem
2) Structure the decision hierarchy
3) Construct a set of pairwise comparison matrices
4) Use the priorities obtained from the comparisons to weigh the priorities in the level immediately below.

AHP provides a consistency ratio (CR) factor, which is used to determine whether participants have answered consistently, i.e. in agreement with themselves, hence gives mathematically rigor for prioritisations [12].

Moreover, identifying critical decisions and performing sensitivity analysis can expose potential issues and lead to an architecture better prepared for future change [13].

As it is neither feasible nor desirable to fully automate the decision making process, semi-formal techniques such as SWOT (Strengths, Weaknesses, Opportunities, and Threats)

could be utilized to treat trade-offs within specific contexts and design drivers [14].

3. Framework for architectures evaluation and selection

This section illustrates a framework for evaluating and selecting different engineering systems architectures through a real case study implementation.

3.1. Case study outline

Engineering Design Systems are complex and evolve according to customers' needs and technological constraints. A well-known challenge is how to achieve continuity and interoperability across legacy engineering systems and modern commercial ones in order to face ever-growing engineering challenges. As a case study, a Framework for evaluating and selecting engineering design system architecture was applied for an engineering company facing such a challenge. Five engineering systems architecture approaches were proposed as follows:

1. Re-write legacy system into a commercial tool through API layer 3-tiers architecture (N-API)
2. Include Adapter layer around the new API layer through 4-tiers architecture (Adapter)
3. Include Translator layer between the legacy system API and a commercial tool through 4-tiers architecture (L-API)
4. Utilise a Domain Specific Language 4-tiers architecture (DSL)
5. Utilise Model Driven Engineering through 4-tiers architecture (MDE)

In order to evaluate and select the most appropriate solution, the framework in Fig. 2 was applied. The process was initiated by SWOT and PEST workshops analysis, capturing internal/external factors to aid the decision making process and enhance stakeholders' understanding of each architecture approach. Quality workshops were held in order to have a deeper understanding of each quality value with respect to each architectural approach.

Quality attributes were used to evaluate the architecture candidates from a pool of architecture approaches using the Analytical Hierarchical Process (AHP) technique.

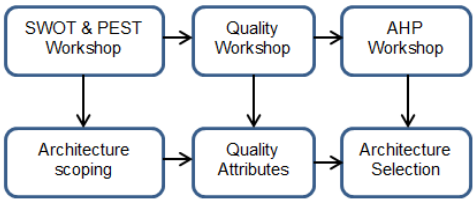

Fig. 2. Architecture evaluation and selection framework

3.2. SWOT and PEST analysis

It is essential at the beginning of the architecture selection process to have a good understanding of proposed architecture approaches within the business's political and market context. An effective way of achieving that is to use SWOT analysis to identify internal and external factors, as well as PEST analysis to identify constraints which should be taken into consideration during the evaluation process. Tables 1 and 2 illustrate PEST and SWOT analyses for the DSL architecture approach.

Table 1. PEST analysis for DSL approach

Political	**Economic**
Use of a DSL has support within IT department	External consultants would likely be required (increased cost)
Social	**Technological**
Developers would spend time learning an approach and toolset that may not be useful outside the company	Additional tools would be required to effectively develop models and the DSL

Table 2. SWOT analysis for DSL approach

Strengths	**Weaknesses**
Change of programming language only requires change to the DSL-to-native translator. No API source code modification is required	System's developers will have low productivity while coming up to speed learning the DSL syntax
Opportunities	**Threats**
The DSL could provide a simplified language syntax vs. the object-oriented APIs	Increased difficulty of integrating the DSL with other components of the IT system

3.3. Identifying architecture quality attributes

The Quality Attributes structure proposed is based on the international standard (ISO/IEC 25010:2011) Systems and software Quality Requirements and Evaluation (SQuaRE).

The first phase is to identify a list of Quality Attributes that are desirable in the system. It was noted through stakeholders' discussions that having a business attribute is desirable for evaluating implementation feasibility (Table 3).

Table 3. Example of utilized quality attributes

Maintainability	**Portability**	**Feasibility**
Modularity	Adaptability	Cost
Reusability	Installability	Schedule
Analysability	Replaceability	
Modifiability		
testability		

For each attribute, scenarios are populated to put the ISO definition within context as shown in the example in Table 4.

Table 4. Populated quality attribute

Quality Attribute	Modularity
Quality Definition	Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components
Quality Scenario	Design engineer will insert new feature into the system without the need to modify existing features

3.4. Analytical Hierarchy Process

The agreed Quality Attributes were used as inputs for the AHP workshops. Pairwise rankings were executed over three levels, with results being recorded in Excel sheets before being translated into the AHP tool.

First level; applying pairwise across the high level quality attributes to define their weights (Fig 3).

Second level; applying pairwise ranking across sub-attribute for each high level quality (Fig. 4).

Third level; applying pairwise comparison across architecture approaches and each quality sub-attribute (Fig. 5).

Attributes	Functional	Performance	Compatibility	Usability	Security	Maintainability	Portability	Feasibility
Functional	1	5	3	1	5	1/5	1/5	7
Performance			1/3	1/5	1/3	1/7	1/5	5
Compatibility			1	1/3	1/2	1/7	1/4	5
Usability				1	5	1/5	1/5	7
Security					1	1/7	1/7	5
Maintainability						1	2	7
Portability							1	7
Feasibility								1

Fig. 3. Pairwise ranking across main quality attributes

Maintainability	modularity	reusability	analysability	modifiability	testability
modularity	1	1/3	2	1/3	1/6
reusability		1	3	1/2	1/5
analysability			1	1/5	1/3
modifiability				1	1/5
testability					1

Fig. 4. Pairwise ranking across sub-quality attributes

Reusability	N_API	Adapter	L_API	DSL	MDE
N_API	1	1/5	3	5	4
Adapter		1	5	7	6
L_API			1	3	2
DSL				1	1/2
MDE					1

Fig. 5. Pairwise ranking of Architectures with each sub-quality attribute

Data recorded was processed using AHP tool (Expert Choice) to check for inconsistencies and build the hierarchy model as shown in Fig. 6.

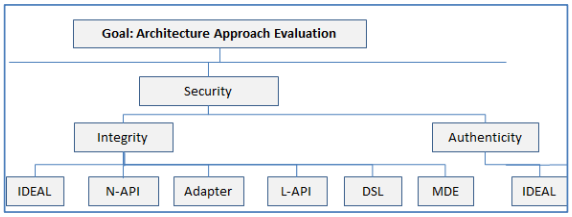

Fig. 6. AHP hierarchy model

As shown in Fig. 7, the highest four attributes came up as; Maintainability (34.2%), Portability (27.0%), Functional and Usability (each 11.6%). The business attribute (Feasibility) was given a marginal weight so that it could be included in the analysis model without disturbing the overall architectures scorings.

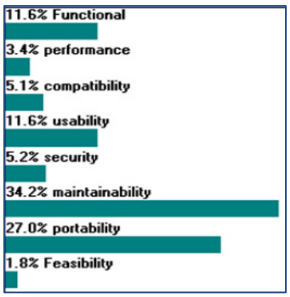

Fig. 7. AHP Dynamic Graph – Attributes scorings

As shown in Fig. 8, DSL approach achieved the top score, which was not surprising as it obtained the highest pairwise ranking against Maintainability and Portability attributes.

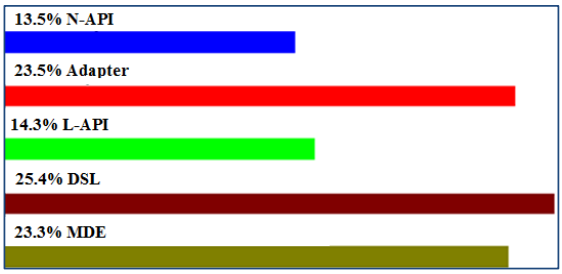

Fig. 8. AHP Dynamic Graph – Architectures scorings

Although the DSL approach came out on the top of the list (25.4%), followed by Adapter layer approach (23.5%), looking into the Feasibility analysis, the Adapter layer approach scored higher than DSL one (26.5% vs 6.4%).

Combining the AHP scores with outcome from SWOT and PEST analyses has given the stakeholders a wider perspective to conclude an informative decision. It was thus agreed that the Adapter layer would strike the right balance between modernising Engineering Design Systems, whilst assuring business continuity in terms of time to market and resources required.

4. Discussion and Conclusion

In this paper, we have illustrated a framework for evaluating alternative Engineering Design System architectures and selecting appropriate one that delivers required quality level, whilst assuring feasibility of implementation. It combines SWOT and PEST analysis with AHP and Quality Attributes trade-offs to provide a wider set of analysis viewpoints. The framework has been validated through application to a real case study.

Quality Attributes were adopted from the ISO standard (ISO/IEC 25010:2011), which enabled more intensive analysis and elicitation of critical characteristics. Some attributes were not utilized as they were not relevant to the system of concern, while new business attributes were adopted to give a more rounded quantitative evaluation viewpoint.

AHP has proven challenging for stakeholders as it forces them to weigh alternatives using pair wise technique. However, putting quantitative scores against attributes and architectures with respect to each other has proven useful in having objectively calculated weights, which increases the confidence of the evaluation process results.

Inconsistency ratio has been beneficial in signaling discrepancies in stakeholders' evaluations. Quite often participants had to revisit their score for revaluation if ratio is inconsistent.

SWOT and PEST analyses allowed the discussion of proposed architectures from different viewpoints, that otherwise may have been missed if stakeholders were to rely

only on Quality Attributes. The selected architecture approach has been approved by subject of matter experts.

General limitation noticed by the researchers was the challenge faced during the Quality Attributes scenarios generation, which was due to the lack of stakeholders who have the experience of system and software architectures qualities attributes, and their implementations' scenarios. This was overcome by involving external consultants during the workshops to enrich the discussions.

Future work identified by the researchers is to study the possibility of applying the framework further to the selected architecture's components' design in order to identify the proper implementation approach.

Acknowledgements

The authors would like to thank the Engineering and Physical Sciences Research Council (EPSRC), the Engineering Doctorate (EngD) Centre at Cranfield University and the engineering company for funding and supporting this research project. The continued support given by all those who give their time for workshops and interviews is also appreciated.

References

[1] Rauch E, Dallasega P, Matt DT. The way from Lean Product Development (LPD) to Smart Product Development (SPD). Procedia CIRP 50, 2016; 26 – 3.

[2] Sheard SA, Mostashari A. Principles of complex systems for systems engineering, Systems Engineering, 2009; vol. 12, no. 4, pp. 295-311.

[3] Nightingale DJ, Rhodes DH, Enterprise systems architecting: Emerging art and science within engineering systems, Proceedings of the ESD External Symposium, Citeseer, 2004.

[4] Mistrik, I, Bahsoon, R, Eeles P, Roshandel R, Stal M. Relating System Quality and Software Architecture. Elsevier, 2015.

[5] Bass L, Clements P, Kazman R. Software Architectures in Practice, 2nd ed. Addison Wesley, Reading, 2003.

[6] Garces L, Ampatzoglou A, Avgeriou P, Nakagawa EY. Quality attributes and quality models for ambient assisted living software systems:A systematic mapping. Journal of Information and Software Technology. 2017; 82:121-138.

[7] Boehm BW, Brown JR, Lipow M. Quantitative evaluation of software quality. ICSE'76: 2nd International Conference on Software Engineering, 1976; 592–605.

[8] ISO/IEC 25010, Systems and software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Software product quality and system quality in use models. 2011

[9] Oquendo F, Leite J, Batista T. Software Architecture in Action. Springer, Switzerlan, 2016.

[10] Barney S, Petersen K, Svahnberg M, Aurum A, Barney H. Software quality trade-offs: A systematic map. Journal of Information and Software Technology. 2012; 54: 651-662.

[11] Saaty TL. Decision making with the analytic hierarchy process. International journal of services sciences, 2008; 1(1): 83-98.

[12] Svahnberg M. An industrial study on building consensus around software architectures and quality attributes. Journal of Software Quality. 2005; 13: 357-375.

[13] Zhu L, Aurum A, Gorton I, Jeffery R. Tradeoff and sensitivity analysis in software architecture evaluation using analysis hierarchy process. Journal of Information and Software Technology. 2004; 46: 805-818.

[14] Zimmermann O, Gschwind T, Kuster J, Leymann F, Schuster N.. Reusable Architectural Decision Models for Enterprise Application Development. International Conference on Quality of Software Architectures, 2007; 15-32.

2017-05-09

Framework for engineering design systems architectures evaluation and selection: case study

Darwish, Mohamed

Elsevier

Mohamed Darwish, Essam Shehab, Framework for Engineering Design Systems Architectures Evaluation and Selection: Case Study, Procedia CIRP, Volume 60, 2017, Pages 128-132
http://dx.doi.org/10.1016/j.procir.2017.01.058